The RStudio IDE

using RStudio to manage code, data, and files

by Martin Frigaard

Written: September 30 2021

Updated: November 24 2021

Created using the "λέξις" theme

The RStudio IDE

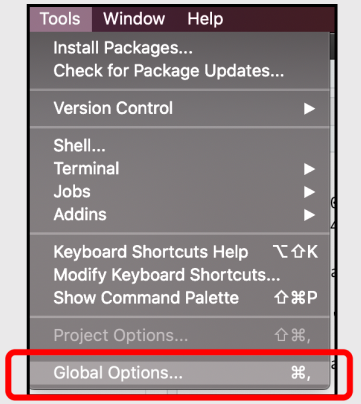
Managing your code, data, and files with RStudio

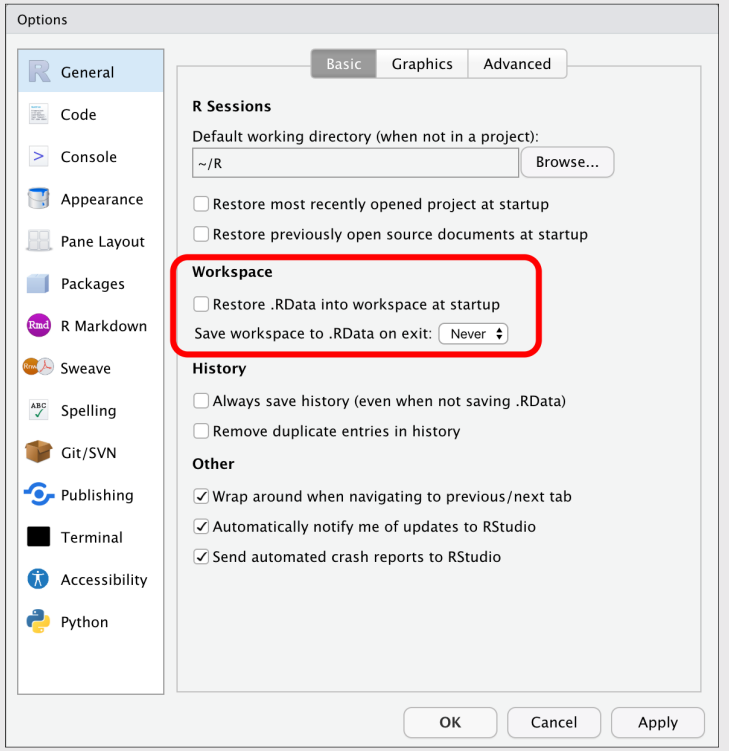
Change RStudio's Default Settings

Click on Tools > Global Options...

- We want to uncheck "*Restore .RData into work space at start up*"

- We also want to make sure we change "*Save work space to .Rdata on exit*" to "*Never*"





Customize RStudio

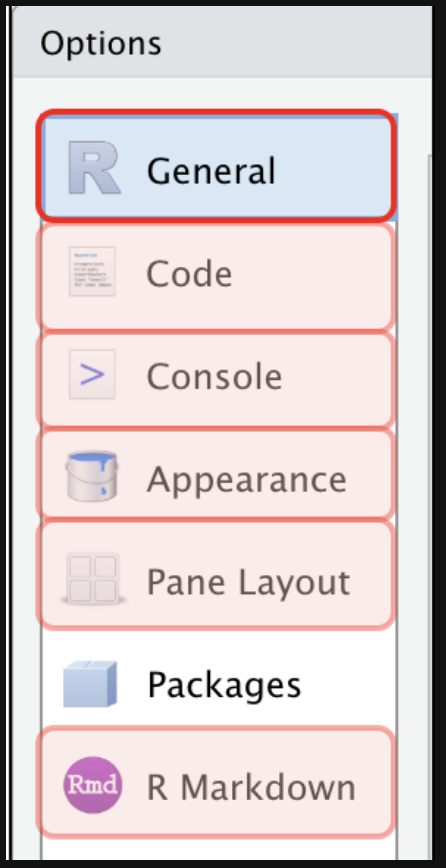
Code

Console

Appearance

Pane Layout

R Markdown



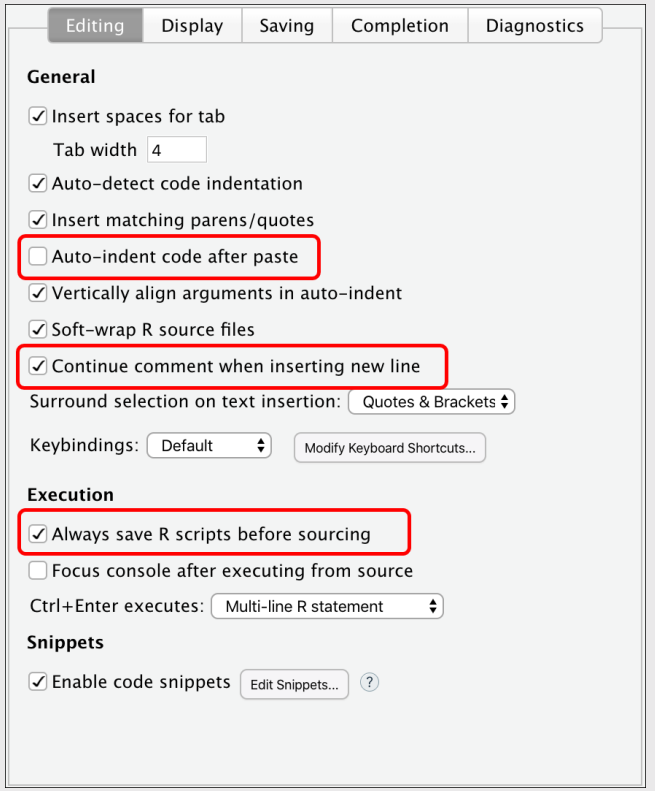
https://github.com/mjfrigaard/csuc-data-journalism

Code Editing

- Auto indent?

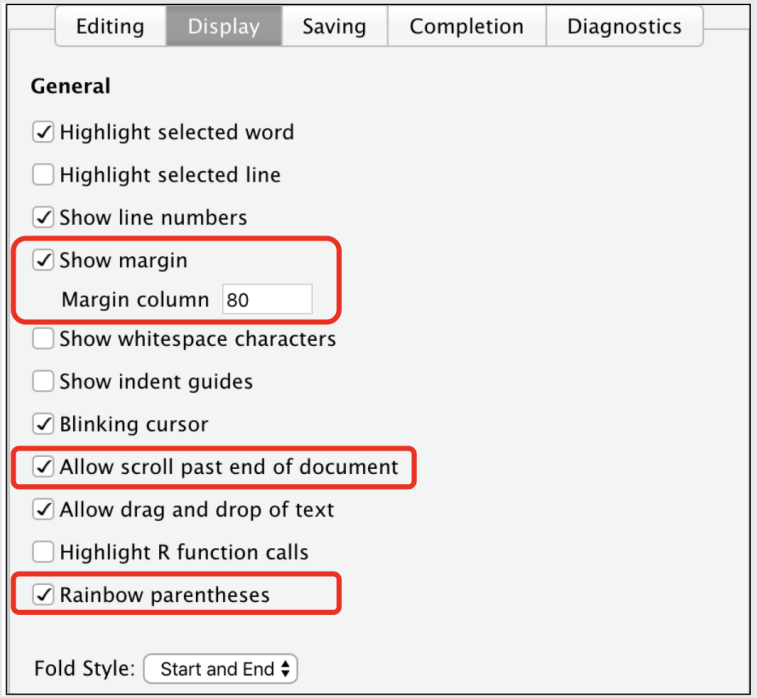
- Continue comment lines?

- Save R scripts before sourcing?



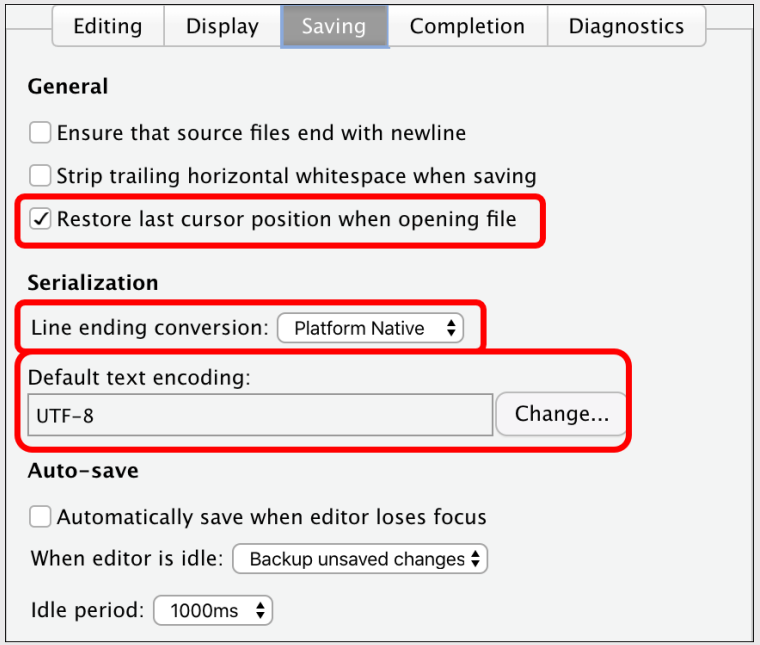
Code Display

- Margins?
- Scrolling?
- Rainbow parentheses?



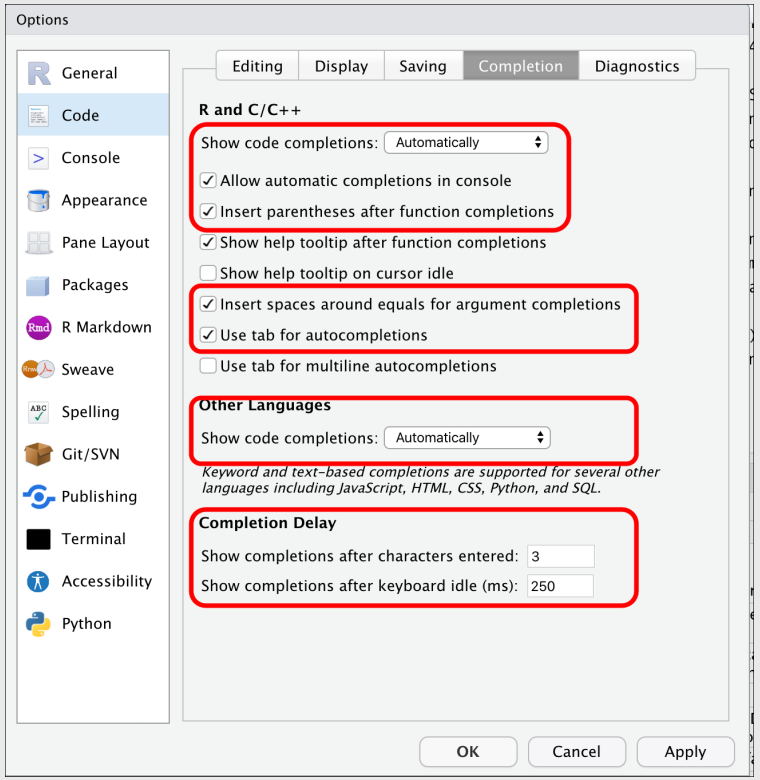
Code Saving

- Cursor position?
- Line endings?
- Text encoding?



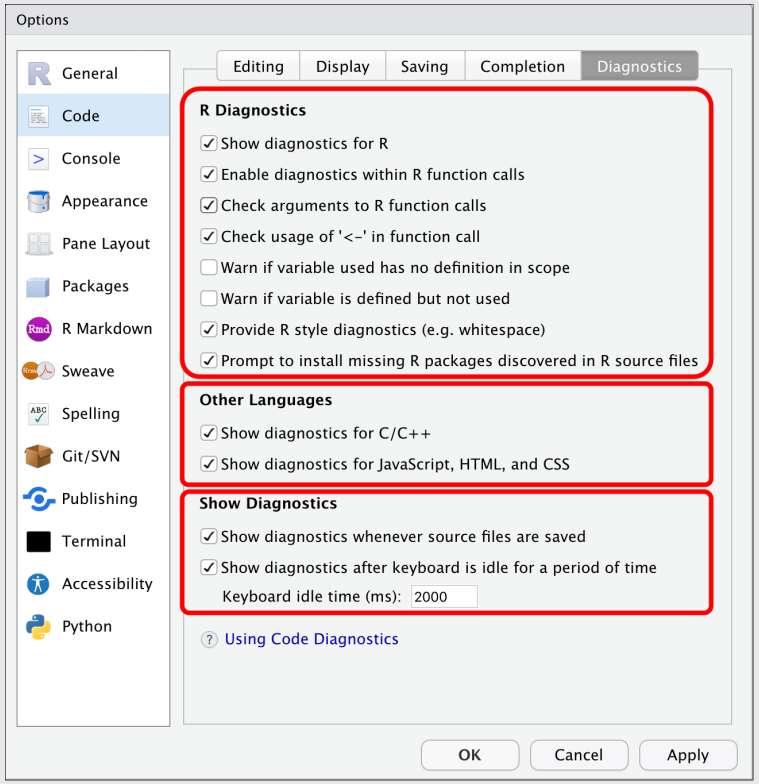
Code Completion

- Insert parentheses?
- Insert spaces?
- Completion delay setting?



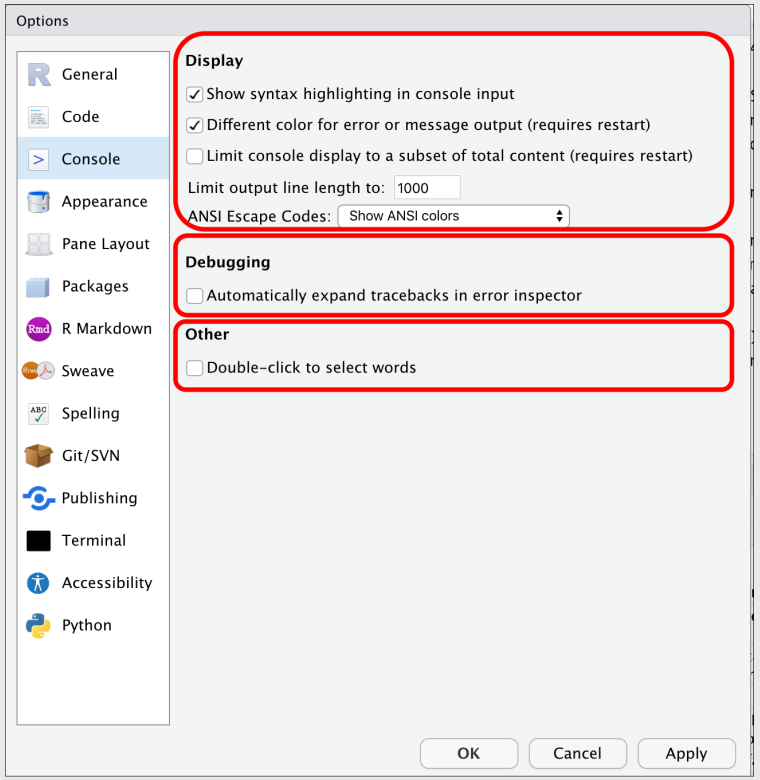
Code Diagnostics

- Check your R Code?
- Check other languages?
- How long?



Console

- Display?
- Debugging?
- Other?



Appearance

- RStudio theme?

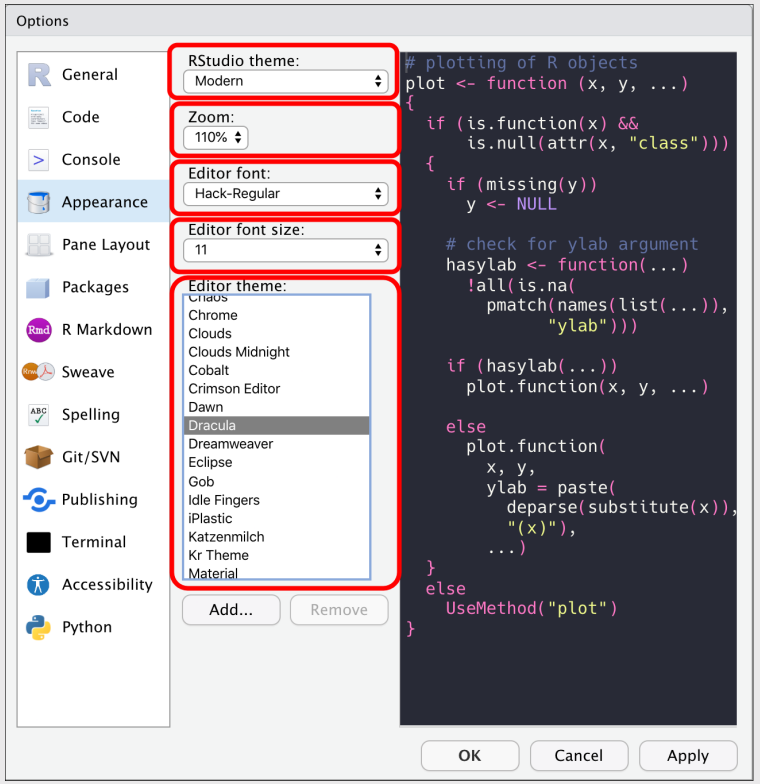
- Zoom?

- Also hold ⌘ and press + on macOS

- Also hold `ctrl` and press + on Windows

- Font?

- Editor theme?



Pane layout

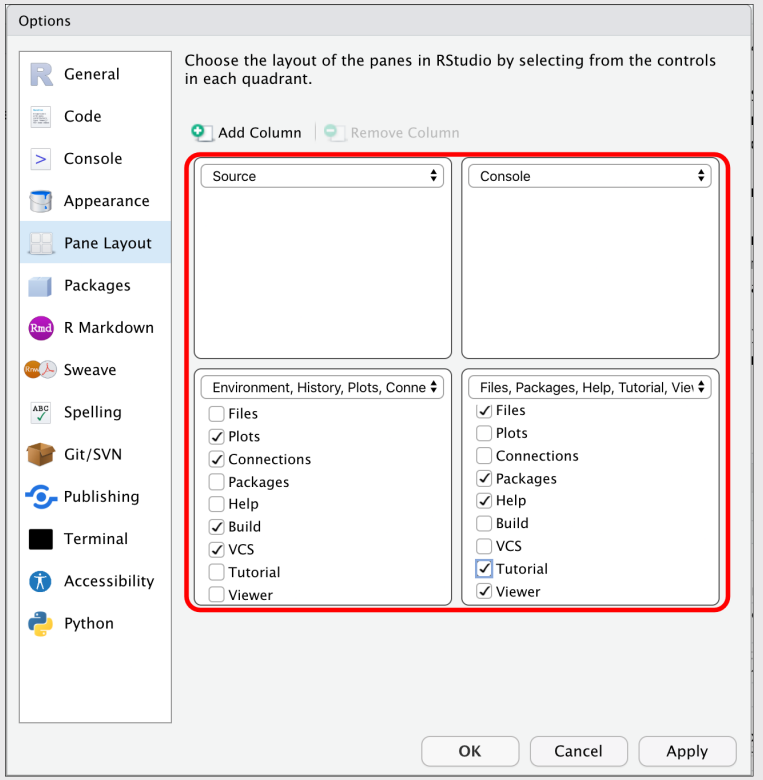
- Source?

- Console?

Combining pane elements?

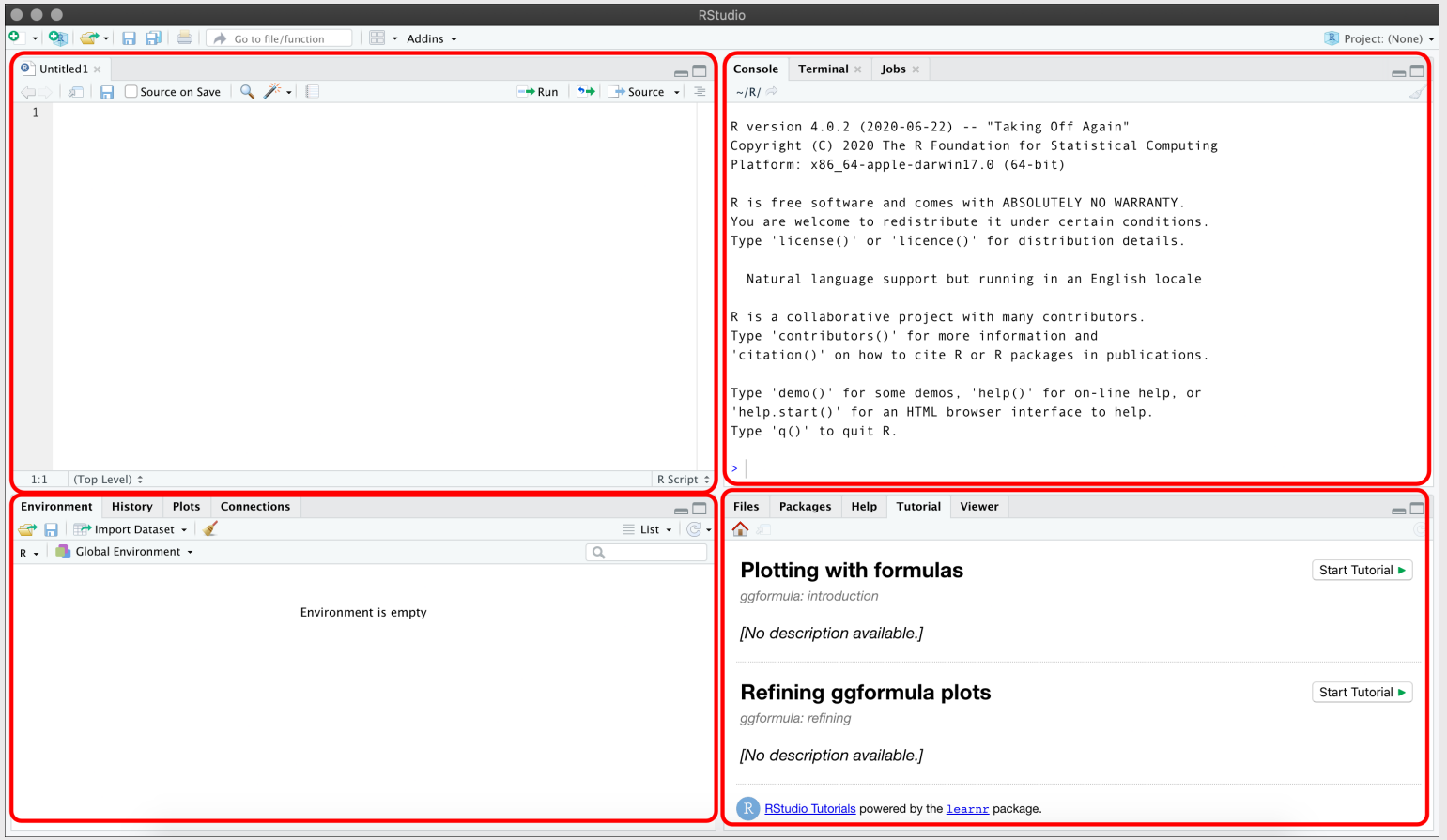
- Plots, Connections, Build, VCS, Presentation

- Files, Packages, Help, Tutorial, Viewer



Pane layout view

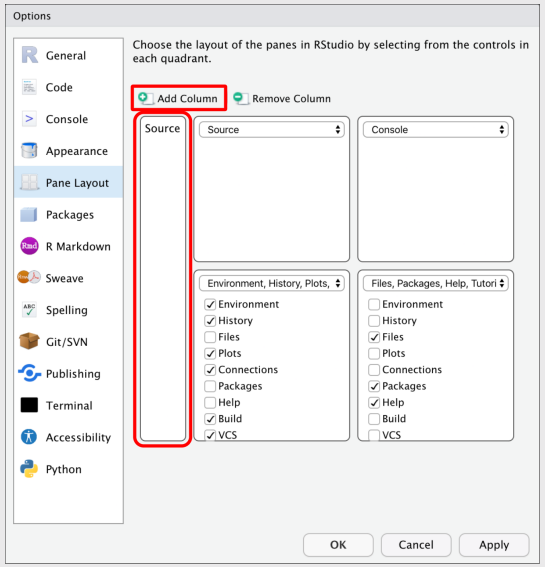
Standard layout options

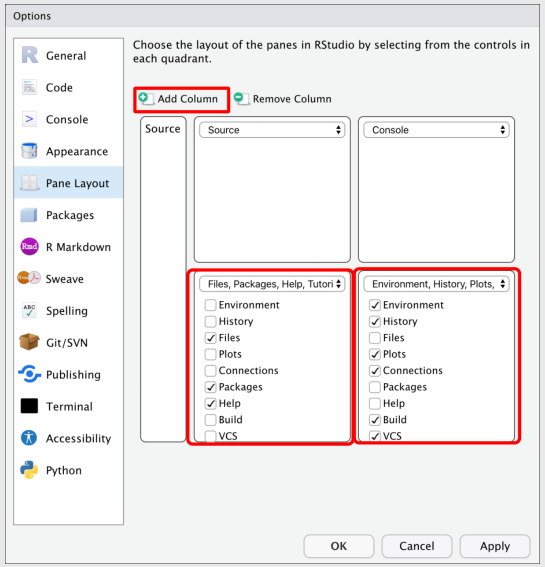


Pane layout: add column

Two screens?

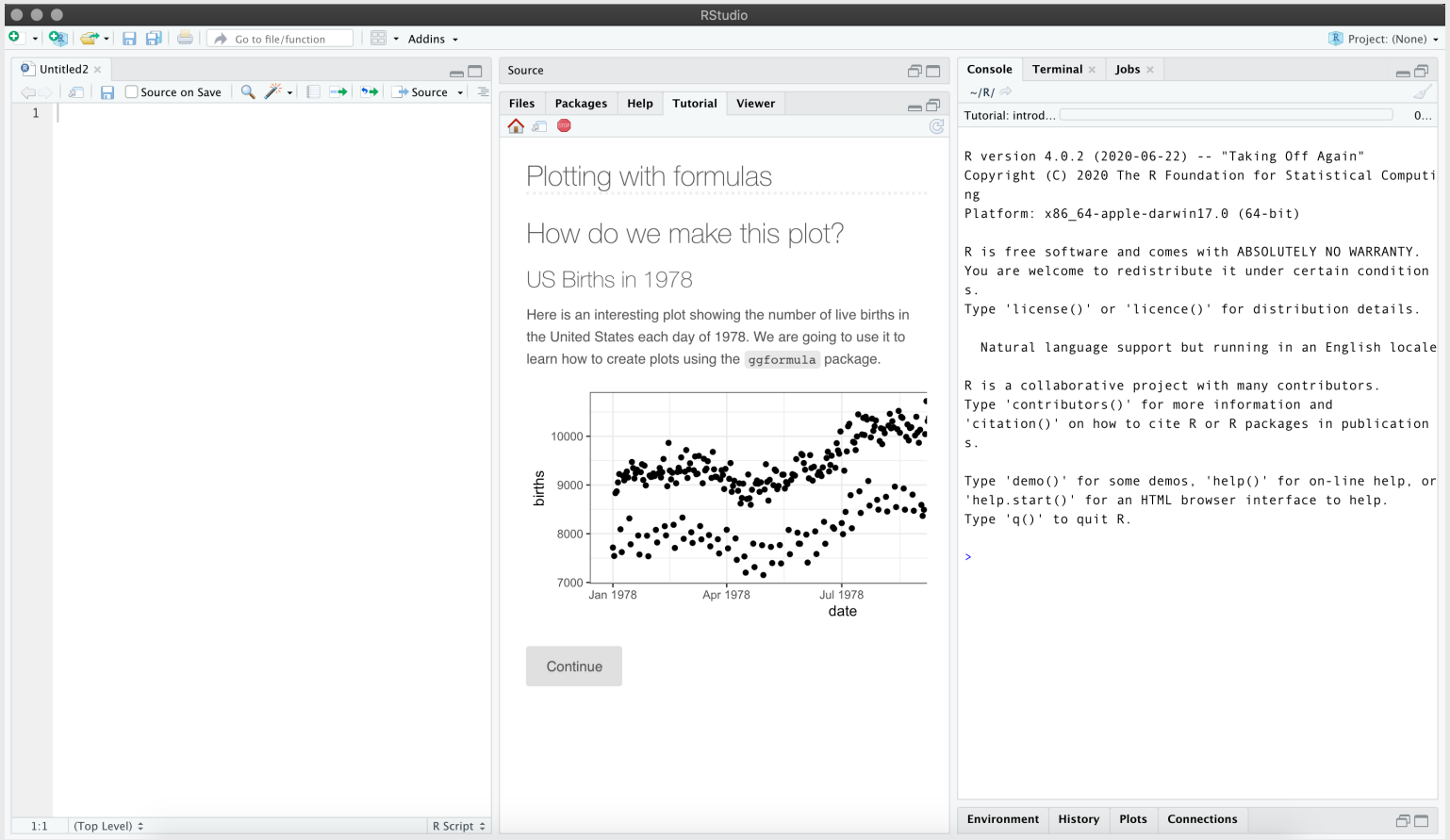
- add a Source column and rearrange the panes





Pane layout: add column view

Now you see **Source**, **Tutorial**, and **Console** panes on a single screen!



RStudio Projects

Why RStudio Projects?

Keep track of all your files with RStudio project files (`.Rproj`).

Self contained

> Using R projects keeps track or your current working directory!

Project orientated

> .Rproj files make bundling and shipping files and folders easier!

Why RStudio Projects?

Keep track of all your files with RStudio project files (`.Rproj`).

Avoid removing all the files

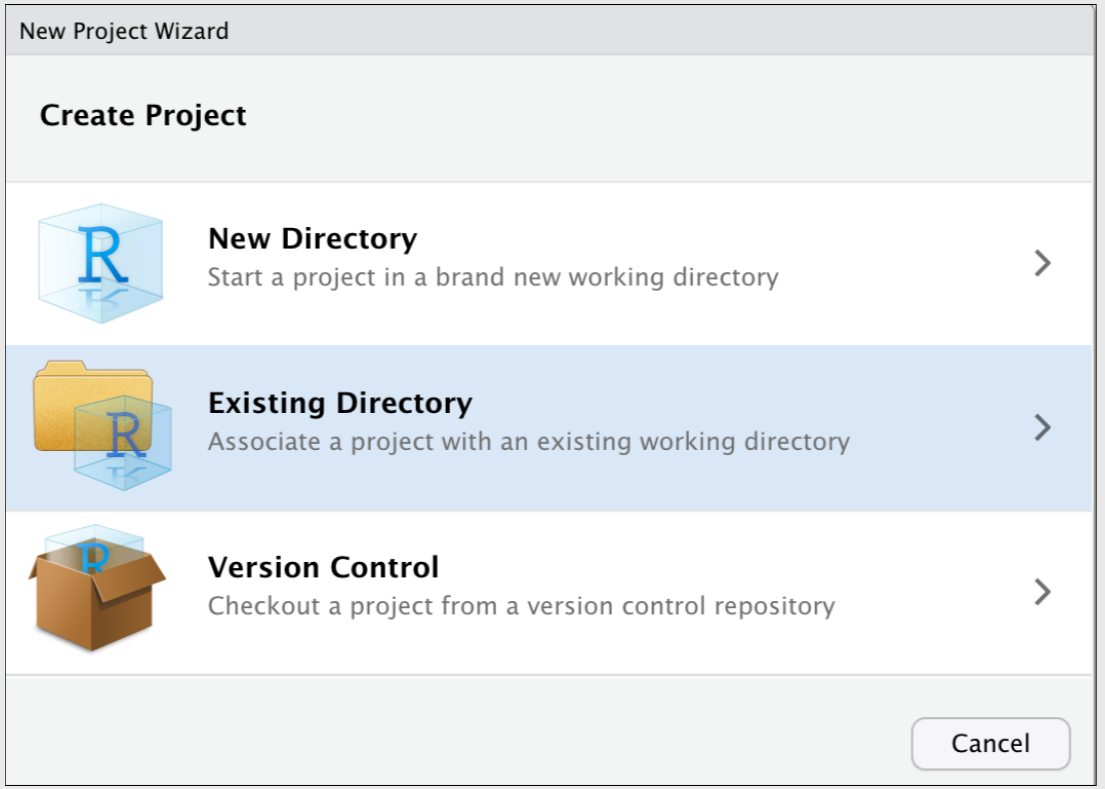
Don't do this!

```
rm(list = ls())
```

`.Rproj` files keep all the files associated with a project together — input data, R scripts, analytic results, figures.

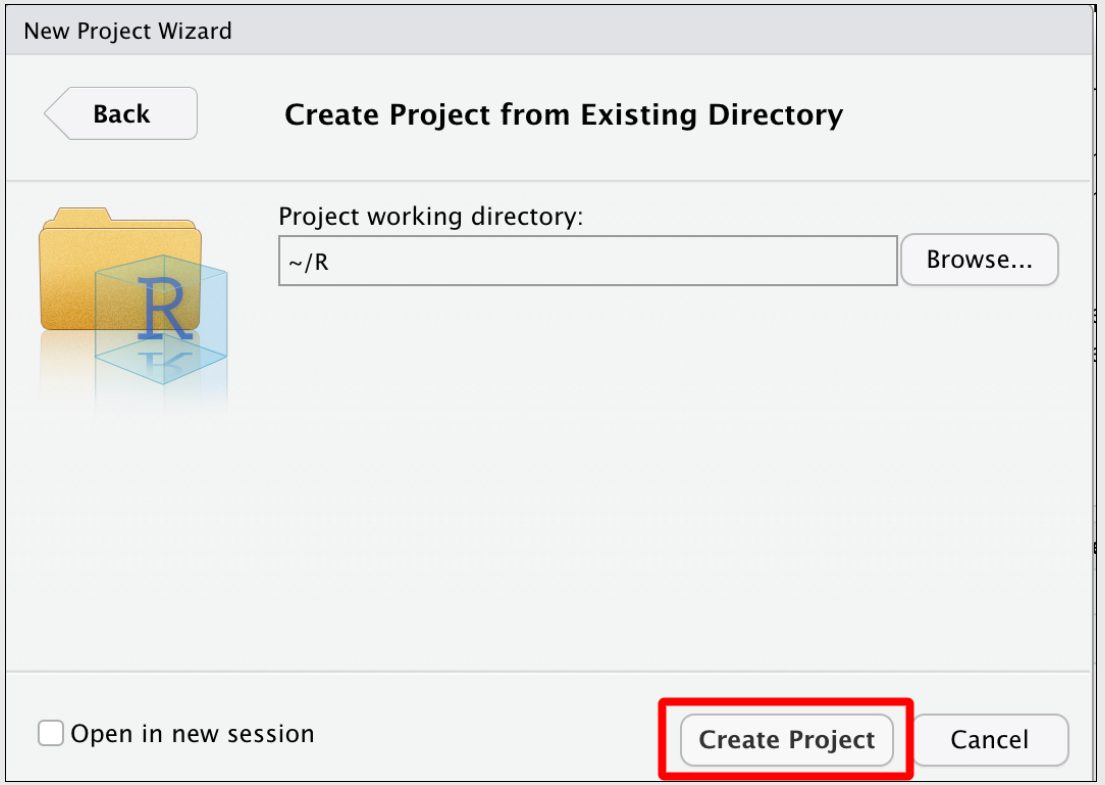
Creating RStudio project in existing folder

Click on *'Project: (None)'* > *'New Project'*



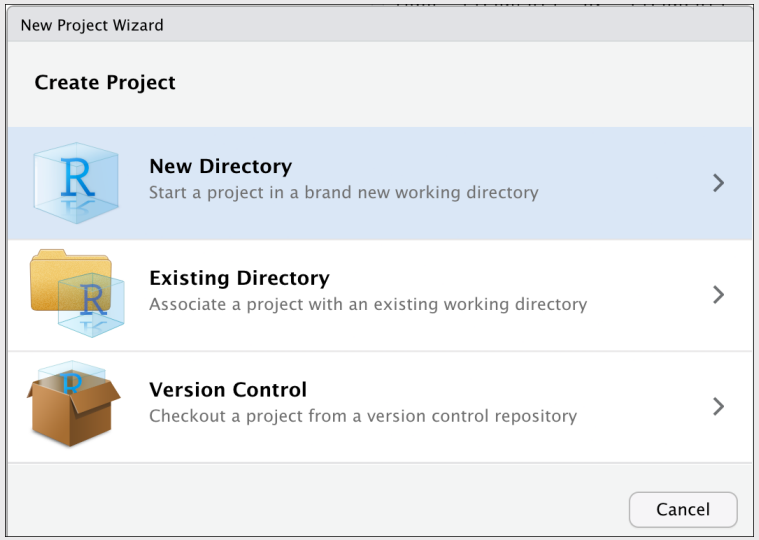
Creating RStudio project in existing folder

Click on *'Browse* > *'Create Project'*

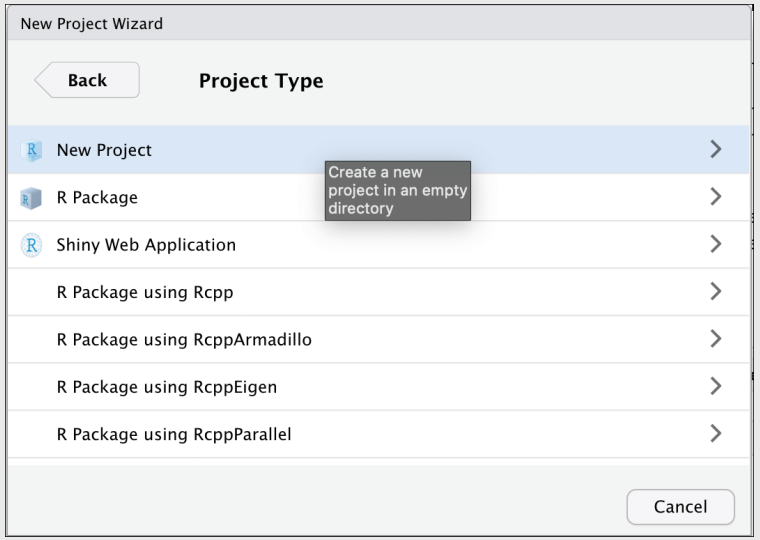


Creating RStudio projects in new folder

Click on Project: (None) > New Project

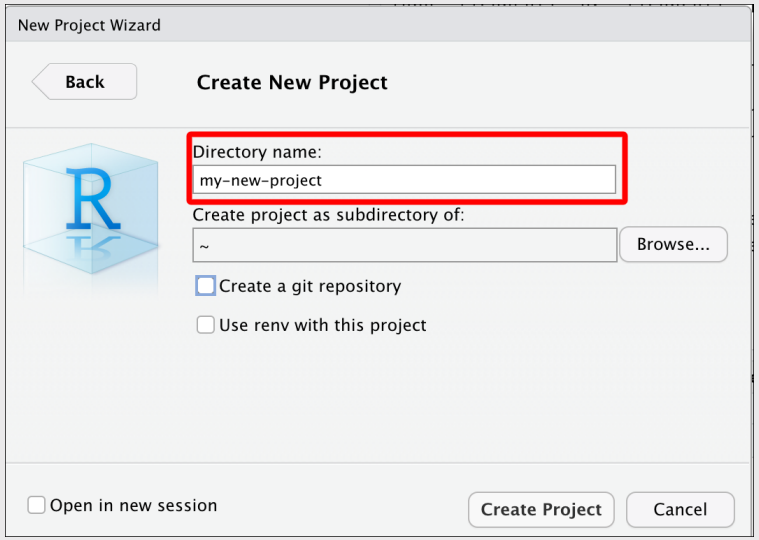


Select project type

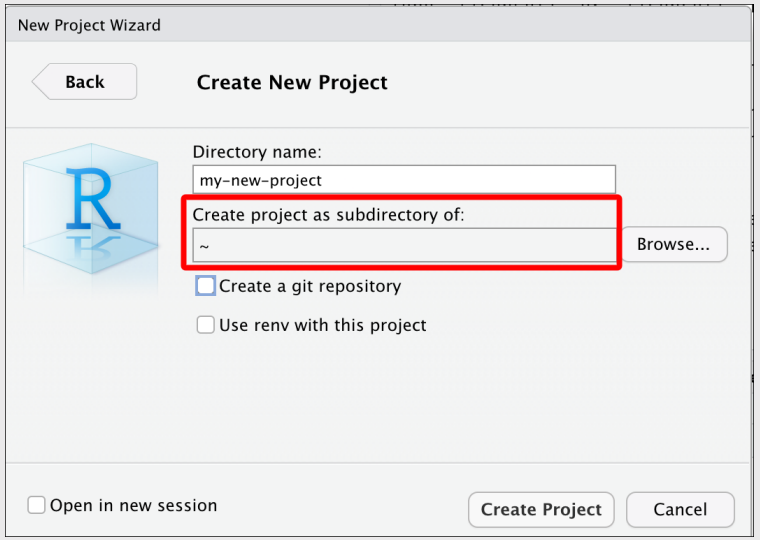


Creating RStudio projects in new folder

Create new folder name

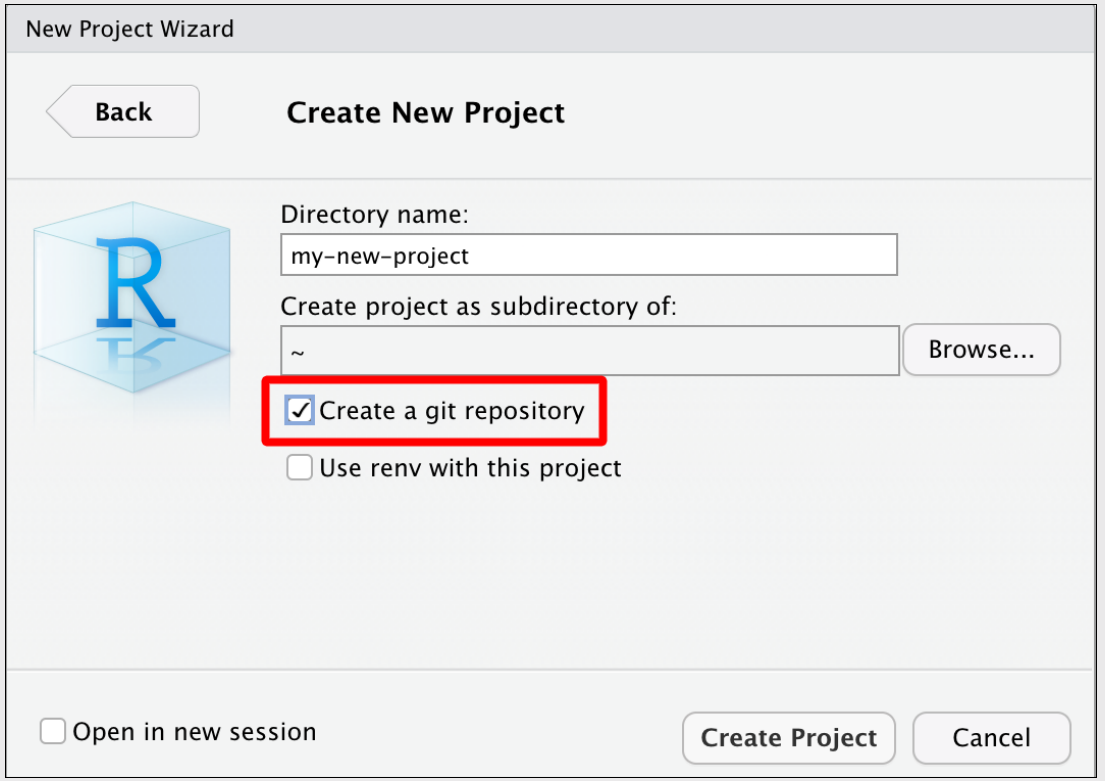


Choose parent folder



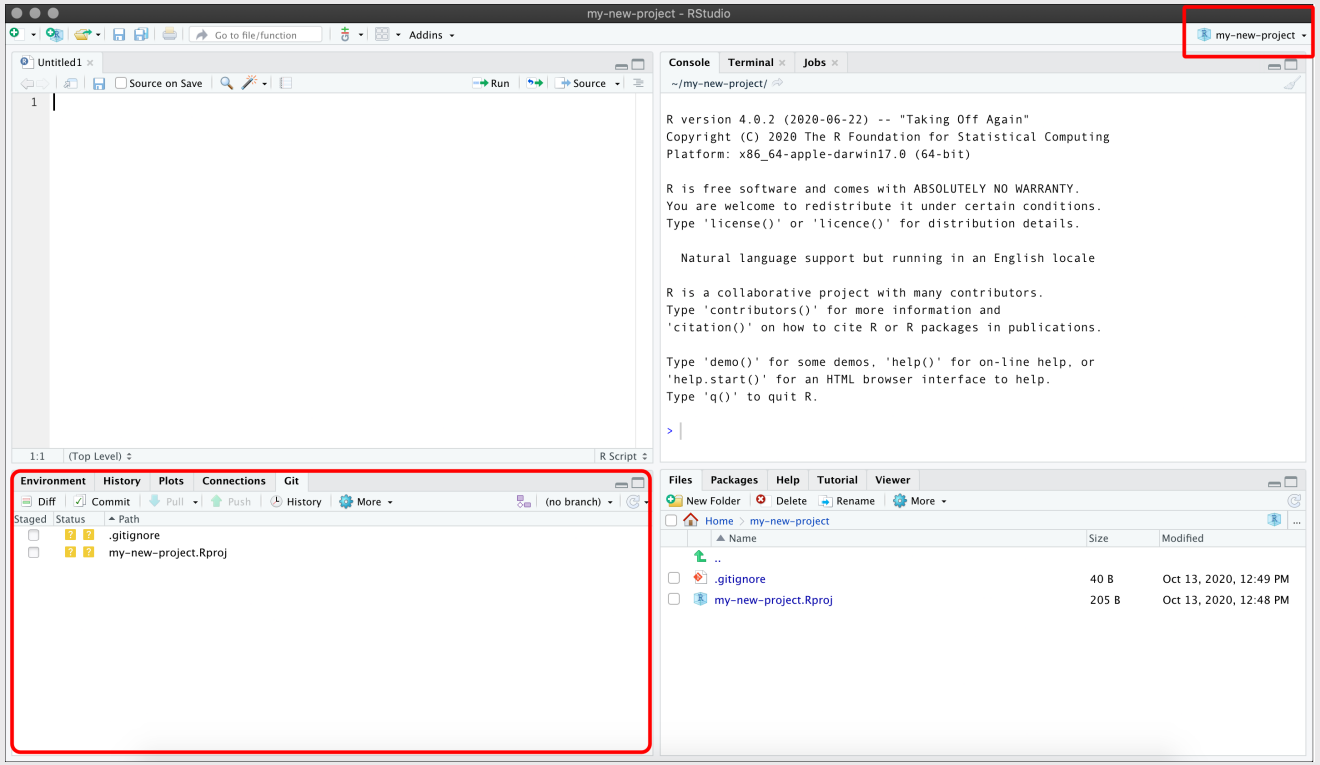
Creating RStudio projects in new folder

If you have Git installed, select *'Create a git repository'*



Creating RStudio projects in new folder

Check for new project name & *Git* pane



Folder Structure

- separate raw and cleaned data
- keep documents and code separate
- keep figures separate
- name files appropriately (preferably 2 digit prefix)
- structure is reusable and easy to understand

https://github.com/mjfrigaard/csuc-data-journalism

```
project-name/
    |-- CITATION
    |-- project-name.Rproj
    |-- README.md
    |-- LICENSE
    |-- requirements.txt
    |--data/
       |--raw/
            |--raw-birds-data.csv
       |--processed/
            |--processed-birds-data.csv
    |--doc/
        |-- notebook.Rmd
        |-- manuscript.Rmd
        |-- changelog.txt
    |--results/
        |-- summarized-results.csv
    |--code/
        |-- 01-sightings-import.R
        |-- 02-sightings-wrangle.R
        |-- 03-sightings-model.R
        |-- runall.R
```

Folder structure

Adapted from from 'Good enough practices in scientific computing'

Naming things

Naming files

File names should be:

1. human readable -> (makes sense)

2. machine readable -> (regex)

3. sort/order well -> (ISO 8601 date)

2020-10-12-270-301- **central-lab-metrics**.csv

2020-10-12- **270-301**-central-lab-metrics.csv

2020-10-12-270-301-central-lab-metrics.csv

Naming files*

We can perform regular expression searches for files like this:

Find 270-301 files

```
grepl(pattern = "270-301",
      x = "2020-10-12-270-301-central-lab-metrics.csv")
```

```
[1] TRUE
```

*Adapted from Jenny Byran's slides

Naming files*

Also acceptable: Logical order and underscores _

```
files
```

```
[1] "01.0_import_lab-data.R"  "02.0_wrangle_lab-data.R" "03.0_eda_lab-data.R"
```

```
stringr::str_split_fixed(string = files, pattern = "_", 3)
```

```
     [,1]   [,2]       [,3]
[1,] "01.0" "import"   "lab-data.R"
[2,] "02.0" "wrangle" "lab-data.R"
[3,] "03.0" "eda"      "lab-data.R"
```

*Adapted from Jenny Byran's slides

File paths

Use relative rather than absolute file paths

Absolute paths are specific to a system

`/project-name/data` → absolute path in macOS

`\\project-name\\data` → absolute path in Windows

Relative paths are specific to a folder

`project-name/data` → relative path in macOS

`project-name\\data` → relative path in Windows

Or use the `here` package

The `here::set_here()` function solves a lot of file path problems (*especially if you're not using R projects*)

```
library(here)
here::set_here(".")
list.files(all.files = TRUE, pattern = "here")
```

```
  [1] ".here"
```

This creates a `.here` file (similar to `.Rproj` files)

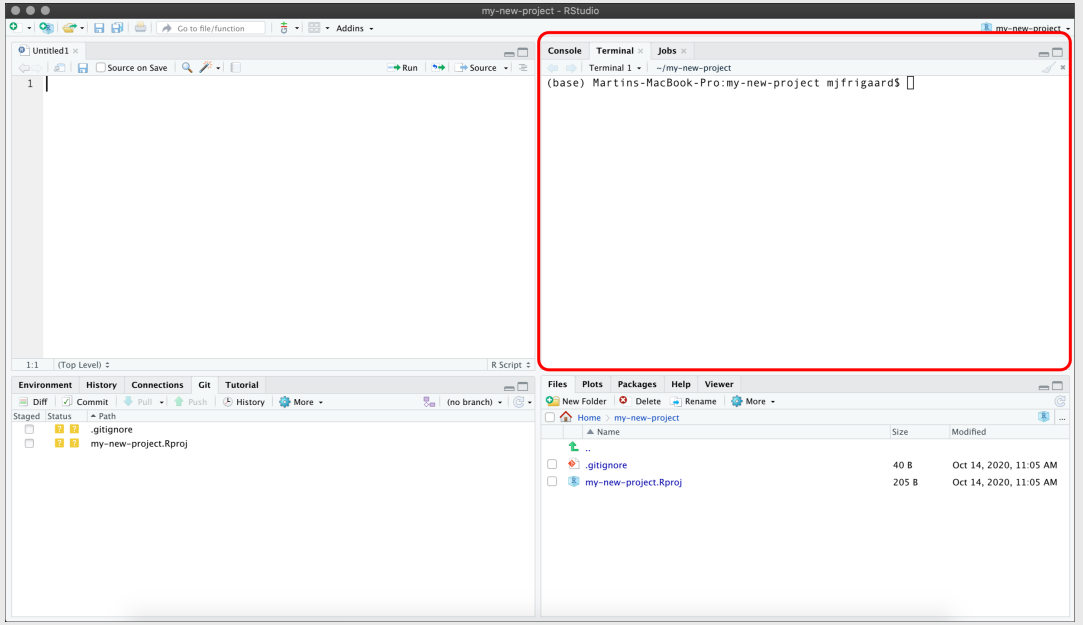
As long as the `.here` file stays in the referenced folder, you can include simply include `here::here()` in the top of your code files.

Terminal pane

Learn a handful of command-line tools to make life easier

`cd`, `pwd`, `mkdir`, `rm`, `ls`, etc.

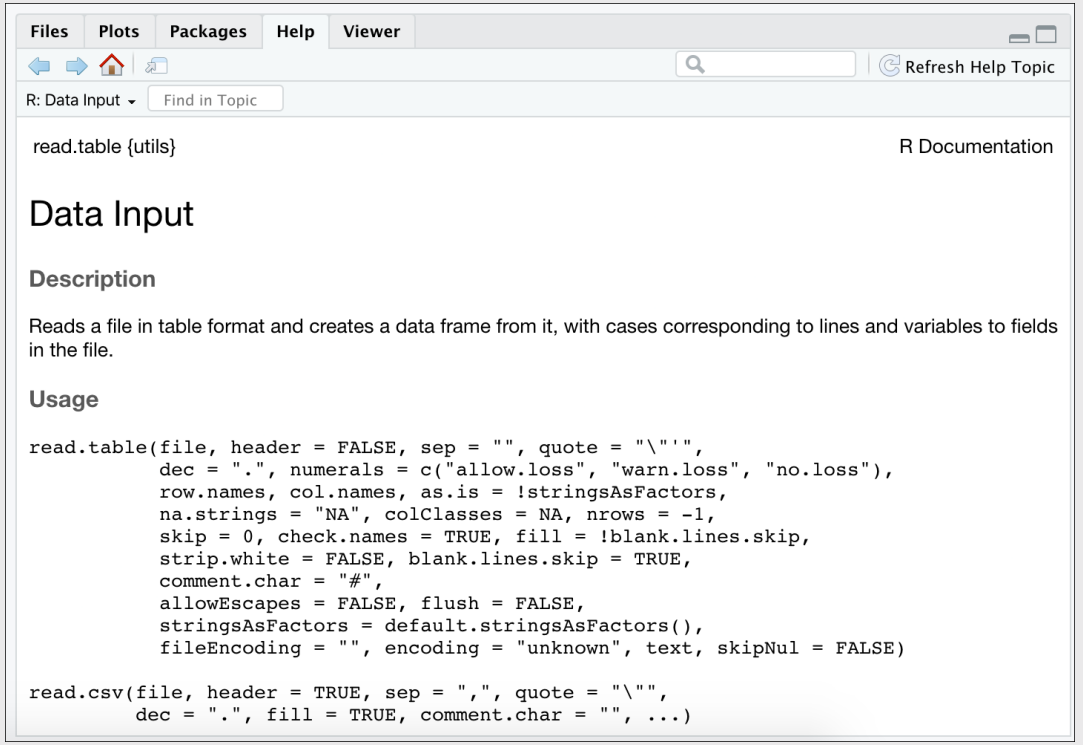
RStudio comes with a Terminal pane for quick access to the command-line



Getting help

R comes with a *ton* of accessible help files

```
?read.csv
```



Getting help online

R also help an incredible community! Click on the links below to see some of the common places for Q & A.

1) Dedicated forum on RStudio Community

2) Questions tagged R on StackOverflow

3) Twitter topics with #rstats hashtag

Asking good questions (reproducible examples)

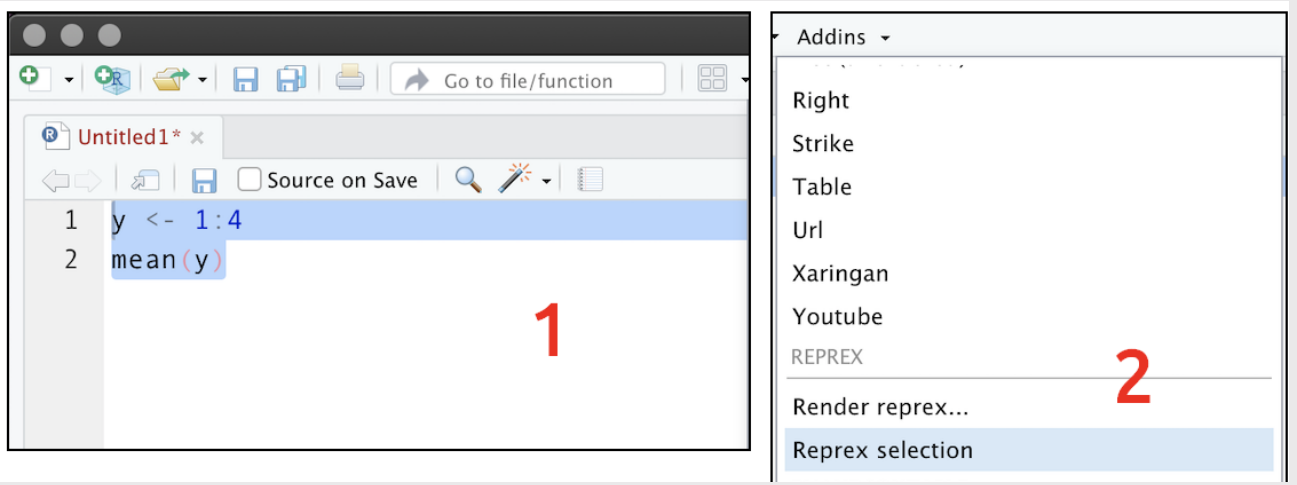
You'll get better results if you ask a question with a reproducible example. The `reprex` package was designed to help you create one!

```
install.packages("reprex")
library(reprex)
```

Use the RStudio Addin to create a reproducible example from code you've copied onto your clipboard!

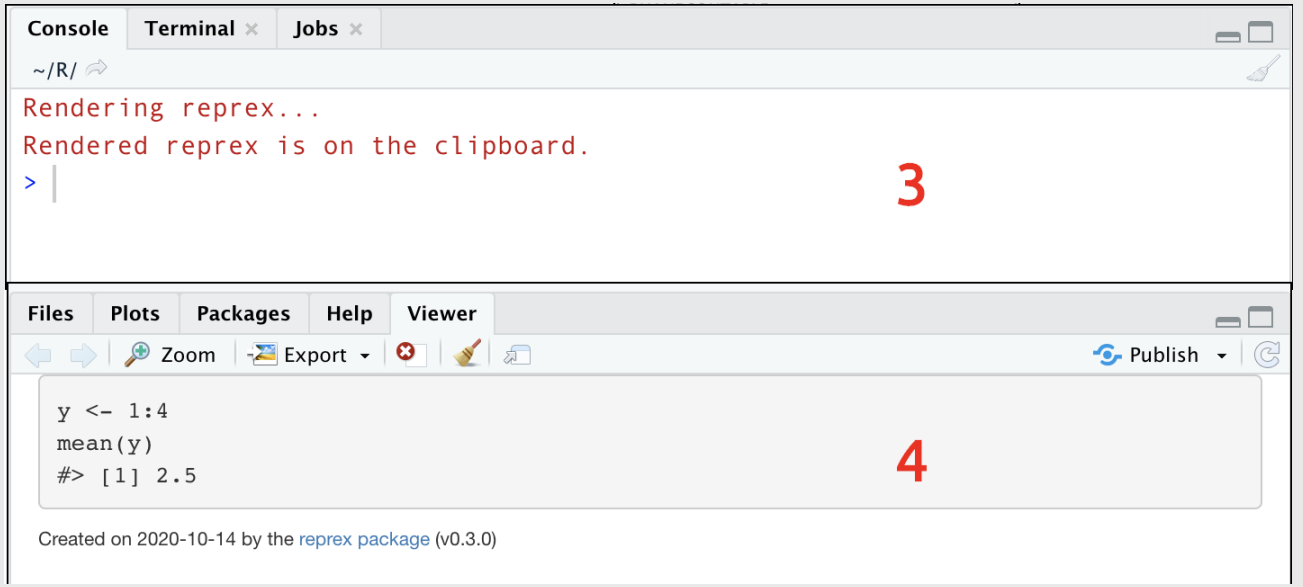
Reprex Addin 1

1. Copy code
2. Select Addin > Render selection



Reprex Addin 2

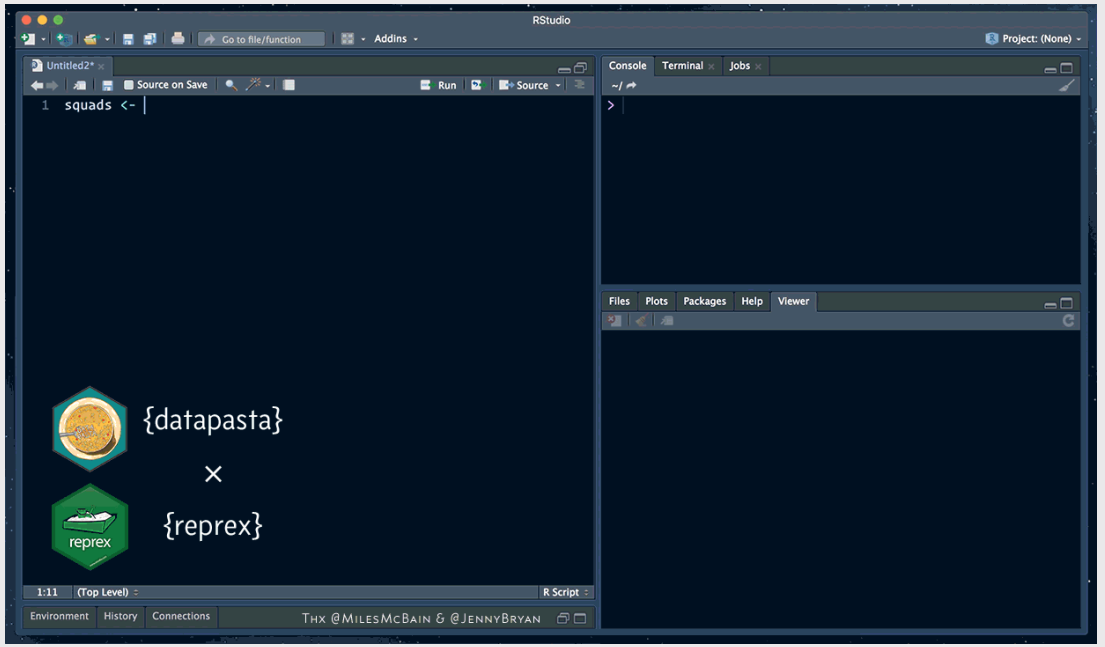
1. Copy code
2. Select Addin > Render selection
3. Wait for console
4. Paste reprex



Reprex + datapasta

To copy + paste actual data in a reproducible example, try `datapasta`!

https://reprex.tidyverse.org/articles/articles/datapasta-reprex.html



Learn more about R best practices:

1. R for Data Science
2. Tidyverse
3. RViews Community Blog

THANK YOU!

Feedback

@mjfrigaard on Twitter and Github

mjfrigaard@gmail.com